UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 10/676,800 | 09/30/2003 | Margaret Ann Bernal | SVL920030038US1 | 2626 |

47069 7590 12/12/2007

KONRAD RAYNES & VICTOR, LLP
ATTN: IBM54
315 SOUTH BEVERLY DRIVE, SUITE 210
BEVERLY HILLS, CA 90212

| EXAMINER | |
|---|---|
| MYINT, DENNIS Y | |

| ART UNIT | PAPER NUMBER |
|---|---|
| 2162 | |

| MAIL DATE | DELIVERY MODE |
|---|---|
| 12/12/2007 | PAPER |

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

| Office Action Summary | Application No. | Applicant(s) | |
|---|---|---|---|
| | 10/676,800 | BERNAL ET AL. | |
| | Examiner | Art Unit | |
| | Dennis Myint | 2162 | |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --*

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

1)☒ Responsive to communication(s) filed on *28 September 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

4)☒ Claim(s) *1-3,5-10,12,14-19,21,23-28,30 and 32-38* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-3, 5-10, 12, 14-19, 21, 23-28, 30, and 32-38* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

**Application Papers**

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

## DETAILED ACTION

*1.* This communication is responsive to Applicant's Amendment, filed on September 28, 2007.

*2.* Claims 1-3, 5-10, 12, 14-19, 21, 23-28, 30, and 32-38 are currently pending in this application. Claims 1, 10, 19, 28, 37, and 38 are independent claims. In the Amendment filed on September 28, 2007, claims 1, 10, 19, 28, 37, and 37 were amended. Claims 2, 4, 11, 13, 20, 22, 29, and 31 have been cancelled. **This office action is made final.**

### *Claim Rejections - 35 USC § 112*

3. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

4. Claims 1-3, 5-10, 12, 14-19, 21, 23-28, 30, and 32-38 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

As per claim 1, the claim in lines 13 recites "wherein the bind-in structure and the statement have the same section number". However, in the steps prior to the limitation ("wherein the bind-in structure and the statement have the same section number"), a (the) statement has never been associated with a section number. Therefore, there is insufficient antecedent basis for this limitation in the claim, that is, there is no

antecedent basis for "a statement with an associated section number" and determining whether the bind-in structure and the statement have a same section number could not be performed.

Claim 1 is therefore rejected under 35 U.S.C. 112 second paragraph. Claims 3, 5, 6, 7, 8, and 9 are rejected 35 U.S.C. 112, second paragraph, because said claims depend on claim 1.

As per claim 10, the claim in lines 13 recites "wherein the bind-in structure and the statement have the same section number". However, in the steps prior to the limitation ("wherein the bind-in structure and the statement have the same section number"), a (the) statement has never been associated with a section number. Therefore, there is insufficient antecedent basis for this limitation in the claim, that is, there is no antecedent basis for "a statement with an associated section number" and determining whether the bind-in structure and the statement have a same section number could not be performed.

Claim 10 is therefore rejected under 35 U.S.C. 112 second paragraph. Claims 12, 14, 15, 16, 17, and 18 are rejected 35 U.S.C. 112, second paragraph, because said claims depend on claim 10.

As per claim 19, the claim in lines 15 recites "wherein the bind-in structure and the statement have the same section number". However, in the steps prior to the limitation ("wherein the bind-in structure and the statement have the same section

number"), a (the) statement has never been associated with a section number. Therefore, there is insufficient antecedent basis for this limitation in the claim, that is, there is no antecedent basis for "a statement with an associated section number" and determining whether the bind-in structure and the statement have a same section number could not be performed.

Claim 19 is therefore rejected under 35 U.S.C. 112 second paragraph. Claims 21, 23, 24, 25, 26, and 27 are rejected 35 U.S.C. 112, second paragraph, because said claims depend on claim 19.

As per claim 28, the claim in lines 15 recites "wherein the bind-in structure and the statement have the same section number". However, in the steps prior to the limitation ("wherein the bind-in structure and the statement have the same section number"), a (the) statement has never been associated with a section number. Therefore, there is insufficient antecedent basis for this limitation in the claim, that is, there is no antecedent basis for "a statement with an associated section number" and determining whether the bind-in structure and the statement have a same section number could not be performed.

Claim 28 is therefore rejected under 35 U.S.C. 112 second paragraph. Claims 30, 32, 33, 34, 35, and 36 are rejected 35 U.S.C. 112, second paragraph, because said claims depend on claim 28.

As per claim 37, the claim in lines 13 recites "wherein the bind-in structure and

the statement have the same section number". However, in the steps prior to the limitation ("wherein the bind-in structure and the statement have the same section number"), a (the) statement has never been associated with a section number. Therefore, there is insufficient antecedent basis for this limitation in the claim, that is, there is no antecedent basis for "a statement with an associated section number" and determining whether the bind-in structure and the statement have a same section number could not be performed.

Claim 37 is therefore rejected under 35 U.S.C. 112 second paragraph.

As per claim 38, the claim in lines 14 recites "wherein the bind-in structure and the statement have the same section number". However, in the steps prior to the limitation ("wherein the bind-in structure and the statement have the same section number"), a (the) statement has never been associated with a section number. Therefore, there is insufficient antecedent basis for this limitation in the claim, that is, there is no antecedent basis for "a statement with an associated section number" and determining whether the bind-in structure and the statement have a same section number could not be performed.

Claim 38 is therefore rejected under 35 U.S.C. 112 second paragraph.

## *Claim Rejections - 35 USC § 101*

5. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

6. Claims 19, 21, 23-27, 28, 30, 32-36, 37, and 38 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

As per claims 19 and 28, the claims in line 1 recite "an article of manufacture". The specification of the instant application defines "an article of manufacture" as " *In such cases, the article of manufacture in which the code is implemented may comprise a transmission media, such as a network transmission line, wireless transmission media, signals propagating through space, radio waves, infrared signals, etc. Thus, the "article of manufacture" may comprise the medium in which the code is embodied*" (Paragraph 0078 of U.S. Patent Application Publication Number 20050071346). "Transmission media, such as a network transmission line, wireless transmission media, signals propagating through space, radio waves, infrared signals, etc" are not statutory and, thus, claims 19 and 28 are rejected under 35 U.S.C. 101 as being non-statutory.

Claims 21 and 23-27 are also rejected under 35 U.S.C. 101 because said claims depends on claim 19. Claims 30 and 32-36 are also rejected under 35 U.S.C. 101 because said claims depend on claim 28.

Claim 37 is rejected under 35 U.S.C. 101 because said claim is directed to software system per se. Said claim recites a system which is software per se according

to Figure 2, Figure 3A, Figure 3B, and Figure 4. As such claim 37 is non-statutory and is rejected under 35 U.S.C. 101.

Claim 38 is rejected under 35 U.S.C. 101 because said claim is directed to software system per se. Said claim recites a system which is software per se according to Figure 2, Figure 3A, Figure 3B, and Figure 4. As such claim 38 is non-statutory and is rejected under 35 U.S.C. 101.

## *Claim Rejections - 35 USC § 103*

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

8. This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

9. Claims 1, 3, 9, 10, 12, 18-21, 27, 28, 30, and 36-38 are rejected under 35 U.S.C. 103(a) as being unpatentable over Agarwal et al., (hereinafter "Agarwal") (U.S. Patent Number 6351742) in view of Kaluskar, U.S. Patent Number 6985904) and further in view of Hayes et al., (hereinafter "Hayes", U.S. Patent Application Number 2002/0129035).

As per claim 1, Agarwal is directed to a method for input parameter binding, when performing bind-in of host variables (Agarwal, Column 4 Lines 8-17, i.e., ***bind variables***; Column 4 Lines 17-25, i.e., *This is not intended to be an exhaustive list of possible descriptions that can be passed to the optimizer. Other descriptions regarding the characteristics, structure, or* ***type of the argument*** *can be used within the scope of the invention. The description of the arguments can be used in the present invention to more effectively estimate selectivity and cost, particularly for database statements containing variables whose values are unknown at compile-time*) and teaches the limitations:

"at bind-time, storing optimization information in a bind-in structure" (Agarwal, Agarwal, Column 4 Lines 8-17, i.e., ***bind variables***. Particularly note that, in the method of Agarwal, binding of bind variables take place either at bind-in or bind-out time. Also note Agarwal, Column 67 through Column 8 Line 22, i.e., *FROM Table3*

*WHERE Table3.col=arctan(:x)*

*This database statement queries for all entries from Table3 in which the values of the column Table3.col equal arctan(:x). Assume that there is no range limitation upon the bind variable ":x". For purposes of this example, the mere fact that :x*

*is a bind variable does not provide further information that is useful for more accurately determining a selectivity value. However, the characteristics of the function "arctan()" does provide useful information that can be used. Specifically, it is known that the output of the arctan() function always produces a value between 0 and 360. Thus, the function or operator itself may have a range constraint (or other useful information) that can be used to calculate* ***selectivity****. The range constraint of the function arctan() is matched against the collected statistics for the Table3.col column to determine the selectivity of the above predicate*); Note that **the collected statistics are optimization information**. Also note that in the said teaching of Agarwal, range constraint and bind variable "x" are used to determine a possible value of x because x could not be known in advance at compile time. In other words, ***the method of Agarwal is comparing data in an application structure received with the statement (i.e., a possible value of x which is in an rage of values;*** **Particularly note that said value is bind-in variable (Agarwal, Column 4 Lines 8-17, i.e.,** ***bind variables)) with the optimization information (collected statistics).*** Said possible value of x is compared to the optimization statistics (i.e., optimization formation), which are already in the in-memory bind-in structure. That the collected statistics are "optimization information" is taught in Column 2 Lines 16-19, i.e., *The cost of an execution plan can be estimated based upon the statistics and selectivity associated with terms within the SQL statement predicate.* In the instant application of the claimed invention, bind-in variables (i.e., data in an application structure received with the statement) are already known in advance). In that case, *the method of Agarwal would compare said bind-in variables (i.e., data in an*

*application structure received with the statement, i.e., variable x which is a bind-in variable) with the optimization information in the memory in bind-in structure*, i.e., collected statistics of optimization in memory);

"wherein the bind-in structure has an associated (section) number" (Agarwal, Column 6 line 64 through column 7 line 5, i.e., *One approach that can be used to address this problem is* ***to associate*** *objects* ***with non-native optimizer-related properties or operations.*** *According to this approach, non-native cost, statistics, selectivity functions are considered* ***object properties*** *that can* ***be associated with*** *various objects on the system,* ***such as*** *for example, user-defined functions,* ***indexes****, indextypes, packages, and columns. If the optimizer encounters an execution plan involving an object which is associated with a non-native optimizer-related function, that function is called to estimate the cost of that execution plan. Further details regarding optimizers and optimizer-related functions (including optimizers directed to non-native objects) are disclosed in co-pending and pending U.S. application Ser. No 09/272,691, titled "METHOD AND MECHANISM FOR EXTENDING NATIVE OPTIMIZATION IN A DATABASE SYSTEM", filed Mar. 18, 1999*). In the preceding teaching of Agarwal, the bind-in structure which holds optimization (**non-native optimizer-related properties or operations)** is associated with "user-defined indexes" (among others) to identify said optimization in the bind-in structure);

"when executing a statement, when performing bind-in of host variables, comparing data in an application structure received with the statement with optimization information in a bind-in structure" (Agarwal, Agarwal, Column 4 Lines 8-17, i.e., ***bind***

***variables***. Particularly note that , in the method of Agarwal, binding of bind variables take place either at bind-in or bind-out time. Also note Agarwal, Column 67 through Column 8 Line 22, i.e., *FROM Table3*

*WHERE Table3.col=arctan(:x)*

*This database statement queries for all entries from Table3 in which the values of the column Table3.col equal arctan(:x). Assume that there is no range limitation upon the bind variable ":x". For purposes of this example, the mere fact that :x is a bind variable does not provide further information that is useful for more accurately determining a selectivity value. However, the characteristics of the function "arctan()" does provide useful information that can be used. Specifically, it is known that the output of the arctan() function always produces a value between 0 and 360. Thus, the function or operator itself may have a range constraint (or other useful information) that can be used to calculate* ***selectivity****. The range constraint of the function arctan() is* ***matched against the collected statistics for the Table3.col column to determine the selectivity of the above predicate***); Note that the collected statistics are optimization information. Also note that in the said teaching of Agarwal, range constraint and **bind variable "x"** are used to determine a possible value of x because x could not be known in advance at compile time. In other words, *the method of Agarwal is comparing data in an application structure received with the statement (i.e., a possible value of x which is in an rage of values;* Particularly note that said value is bind-in variable (Agarwal, Column 4 Lines 8-17, i.e., ***bind variables))*** *with the optimization information (collected statistics).* Said possible value of x is compared to the

optimization statistics (i.e., optimization formation), which are already in the in-memory bind-in structure. That the collected statistics are "optimization information" is taught in Column 2 Lines 16-19, i.e., *The cost of an execution plan can be estimated based upon the statistics and selectivity associated with terms within the SQL statement predicate.* In the instant application of the claimed invention, bind-in variables (i.e., data in an application structure received with the statement) are already known in advance. In that case, the method of Agarwal would compare said bind-in variables (i.e., data in an application structure received with the statement, i.e., variable x which is a bind-in variable) with the optimization information in the memory in bind-in structure.)

"wherein the optimization information includes **at least one** of **data type**,( length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid)" (Agarwal, Column 4 Lines 6-26, i.e., *A second category of information passed to the optimizer involves a description of the arguments in the database statement (104). For example,* ***the argument type*** *for each argument in the database statement can be passed to the optimizer. The following are examples of such argument types: literals,* ***bind variables****, columns,* ***type attributes****, NULL , none of the above. This is not intended to be an exhaustive list of possible descriptions that can be passed to the optimizer. Other descriptions regarding the characteristics, structure, or type of the argument can be used within the scope of the invention. The description of the arguments can be used in the present invention to more effectively estimate selectivity and cost, particularly for database statements containing variables whose values are unknown at compile-time;*

Agarwal Column 5 Lines 50-61, i.e., *This database statement queries for all entries from table emp_table where the value of column "Table2.col" equals the bind variable :x.* ***The arguments to the "equal()" function are passed to the optimizer*** *(i.e., the "Table2.col" and ":x" arguments).* ***Since :x is a bind variable whose value is unknown at compile-time****, the present invention further passes a description of the arguments to the optimizer. For example,* ***the argument "type" of the arguments can be passed to the optimizer****. The argument type of the Table2.col argument is "column" and the argument type of the :x argument is "bind variable"*)

"when there is a match between the data in the application structure and data in the optimization information in the bind-in structure, executing the statement with the optimization information" (Agarwal, Column 4 Lines 41-44, i.e., *The optimizer then selects for execution plan having the lowest relative cost*) "to perform one of fetching data from the data store (and inserting data into the data store) (Agarwal, Column 4 Line 41 through Column 5 Line 45, i.e., *As an example, consider the following databaes statement:*

***SELECT ****

*FROM Table1 , table"* (column 4 lines 45-47) ).

Agarwal's examples teach selecting/fetching data from a data store by way SQL "select" statements. Agrawal explicitly taught "associating bind-in structures with user-defined indexes". Agrawal additionally taught matching the data in the application structure and the optimization information as discussed in detail above.

Agarwal does not explicitly teach the limitations:

"(the bind-in structure has an associated) section number" (Again, it is pointed out that Agarwal teach associating bind-in structures with user-defined indexes but not with "section number"); "wherein the bind-in structure and the statement have a same section number", "where there is not a match (between the data in the application structure and the optimization information)", "regenerating optimization information", and "executing the statement with the regenerated optimization information to perform one of fetching data from the data store and inserting data into the data store". Note that the limitation(s) in the parentheses are taught by Agarwal as discussed above.

On the other hand, Kaluskar teaches the limitation:

"when there is not a match between the currently-issued SQL statement and (in the application structure) and the optimization information" (Kaluskar, Column 3 Line 57-64, i.e., ***If a match is not found***, *then compilation proceeds as in Fig. 1* ; Also note Kaluskar Column 3 Lines 15-25, i.e., *The result of a hard parse is* ***a memory-resident data structure***, *which dictates to the server, by way of the execution plan,* ***the best method for carrying out the database statement request***. *A cursor is one example of such a data structure that is essentially a handle to a memory location where the details and results of a parsed and optimized database statement reside. The cursor comprises, among other things, a compiled SQL statement with its associated execution plan;* Kaluskar, Column 3 Line 39-45, i.e., *The systems and methods for sharing of execution plans for similar SQL statements overcome the disadvantages discussed above* ***by reusing the execution plan*** *of an existing cursor in those situations* ***where a client issues a SQL statement similar to another SQL statement previously***

***compiled*** *;* Kaluskar, column 3 Line 51-55, i.e., *Step 205 computes* ***a hash value*** *to determine* ***if a matching SQL statement exists in the shared memory pool****. If a match is found, step 210, then the performance penalty of initiating a hard parse can be avoided altogether--the existing plan compiled for the matching SQL statement will be reused for the instant SQL statement (step 220). If a match is not found, then compilation proceeds as in FIG. 1 (step 215), beginning with parse phase 150 and ending with execution plan creation 130.* Note that Kaluskar's method is employs "a hash" value to match the currently issued SQL statement with the SQL statement that have already been compiled, and has been associated with a cursor which points to the best execution plan for that existing (i.e., already compiled) SQL optimization information )

"regenerating optimization information"" (Kaluskar, Column 3 Line 57-64, i.e., *If a match is not found, then compilation proceeds as in FIG. 1 (step 215), beginning with parse phase 150 and* ***ending with execution plan creation 130***);

"executing the statement with the regenerated optimization information to perform one of fetching data from the data store and inserting data into the data store" (Kaluskar, Figure 2, i.e., 210 –N→ COMPILE CURSOR AND CREATE EXECUTION PLAN (Figure 1) ---→ the point after REUSE CURSOR 220; Kaluskar, Column 3 Line 39-45, i.e., *The systems and methods for sharing of execution plans for similar SQL statements overcome the disadvantages discussed above* ***by reusing the execution plan*** *of an existing cursor in those situations where a client issues a SQL statement similar to another SQL statement previously compiled.* Particularly note the recursive

loop of execution in the method of Kaluskar, that is, if there is not optimization information (i.e., execution plan) for a particular SQL and related SQL data, a new execution plan with optimization is regenerated for use later for executing the statement). Note that SQL statements are for fetching (SQL command: SELECT), updating (SQL command: UPDATE), and inserting (SQL command: INSERT) to fetch data from or store data into data stores or update or change data in data stores, which is notoriously well known to everyone in the art.

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to modify the method of Agarwal to add the feature of regeneration optimization information when there is not a match between the data (in application structure) and the optimization, as taught by Kaluskar, so that the resultant method (i.e., the combination of Agarwal and Kaluskar ) would teach regenerating optimization when there is not a match between the data in the application structure and the optimization information. One would have been motivated to do so in order to accomplishment some performance enhancement (Kaluskar, Column 1 Lines 37-39).

Agarwal in view of Kaluskar does not explicitly teach the limitations: "(the bind-in structure has an associated) section number" (Again, it is pointed out that Agarwal teach associating bind-in structures with user-defined indexes but not with "section number"); "wherein the bind-in structure and the statement have a same section number".

On the other hand, Hayes teaches "associating a section number to a component of a database section" (wherein said database section component could be SQL

statements (both static and dynamic), total sorts of output data, and the like) (Hayes, Figure 12 and Paragraph 0063, i.e., *In EqualizeStatement, Global Application List, AL, is first initialized to NULL. Each entry of AL list contains a list of Packages, PKGS, which is initialized to NULL: Each entry of PKGS list contains a list of Section Numbers, SN, which is initialized to NULL.* ***If the Statement type is Static, the Section Number uniquely identifies the Statement****. If the Statement type is Dynamic, different statements (syntactically) may share the same Section Number).*

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to modify the method of Agarwal in view of Kaluskar to add the feature of "associating a section number to a component of a database section", as taught by Hayes, so that in the resultant method (i.e., the combined method of Agarwal in view of Kaluskar and further in view of Hayes), both "the bind-in structure and (the) statement ***(both of which belongs to the same database section)*** would be associated with the "same section number". As such, Agarwal in view of Kaluskar and further in view of Hayes would teach the limitations: "the bind-in structure has an associated) section number" and "wherein the bind-in structure and the statement have a same section number". One would have been motivated to do so in order to *provide an effective performance monitoring and management system"* (Hayes, Paragraph 0007).

As per claim 3, Agarwal in view of Kaluskar and further in view of Hayes teaches the limitation:

"at bind time, storing the optimization information in the bind-in structure" (Agarwal, Column 4 Lines 1-4, i.e, *According to this embodiment of the invention, at least two categories of information regarding variables or arguments in a database statement can be passed to an optimizer;* Column 4 Lines 39-42, i.e., *The estimated cost may be generated by the use of the previously calculated selectivity value*).

Referring to claim 9, Agarwal in view of Kaluskar and further in view of Hayes teaches the limitation:

"when returning a handle to a cursor to a result set from a stored procedure to an application, recalculating the optimization information" (Kaluskar, Column 3 Line 57-64, i.e., *If a match is not found, then compilation proceeds as in Fig. 1*). Note that when a stored procedure returns a cursor to a result set to an application, said cursor will be a new cursor in the method and system of Kaluskar in view of Crone and it would have been calculated/generated as new following the conventional way of hard parsing. In the specification of this application, such situation arises when the caller of the stored procedure is a distributed application, which does not provide a SQLDA (Specification, Paragraph 0074).

As per claim 10, Agarwal in view of Kaluskar and further in view of Hayes teaches the limitations:

""at bind-time, storing optimization information in a bind-out structure" (Agarwal, Agarwal, Column 4 Lines 8-17, i.e., ***bind variables***. Particularly note that, in the method of Agarwal, binding of bind variables take place either at bind-in or bind-out time. Also note Agarwal, Column 67 through Column 8 Line 22, i.e., *FROM Table3*

*WHERE Table3.col=arctan(:x)*

*This database statement queries for all entries from Table3 in which the values of the column Table3.col equal arctan(:x). Assume that there is no range limitation upon the bind variable ":x". For purposes of this example, the mere fact that :x is a bind variable does not provide further information that is useful for more accurately determining a selectivity value. However, the characteristics of the function "arctan()" does provide useful information that can be used. Specifically, it is known that the output of the arctan() function always produces a value between 0 and 360. Thus, the function or operator itself may have a range constraint (or other useful information) that can be used to calculate* ***selectivity****. The range constraint of the function arctan() is matched against the collected statistics for the Table3.col column to determine the selectivity of the above predicate*); Note that **the collected statistics are optimization information**. Also note that in the said teaching of Agarwal, range constraint and bind variable "x" are used to determine a possible value of x because x could not be known in advance at compile time. In other words, ***the method of Agarwal is comparing data in an application structure received with the statement (i.e., a possible value of x***

***which is in an rage of values;*** **Particularly note that said value is bind-in variable (Agarwal, Column 4 Lines 8-17, i.e., *bind variables)) with the optimization information (collected statistics).*** Said possible value of x is compared to the optimization statistics (i.e., optimization formation), which are already in the in-memory bind-in structure. That the collected statistics are "optimization information" is taught in Column 2 Lines 16-19, i.e., *The cost of an execution plan can be estimated based upon the statistics and selectivity associated with terms within the SQL statement predicate.* In the instant application of the claimed invention, bind-in variables (i.e., data in an application structure received with the statement) are already known in advance). In that case, *the method of Agarwal would compare said bind-in variables (i.e., data in an application structure received with the statement, i.e., variable x which is a bind-in variable) with the optimization information in the memory in bind-in structure*, i.e., collected statistics of optimization in memory);

"wherein the bind-in structure has an associated section number" (Agarwal in view of Kaluskar and further in view of Hayes as discussed with respect to claim 1 above);

"when executing a statement, when performing bind-out of host variables, comparing data in an application structure received with the statement with optimization information in a bind-in structure" (Agarwal, Agarwal, Column 4 Lines 8-17, i.e., ***bind variables***. Particularly note that , in the method of Agarwal, binding of bind variables take place either at bind-in or bind-out time. Also note Agarwal, Column 67 through Column 8 Line 22 as applied to claim 1 and discussed in detail with respect to claim 1);

"wherein the optimization information includes **at least one** of **data type**,( length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid)" (Agarwal, Column 4 Lines 6-26, i.e., *A second category of information passed to the optimizer involves a description of the arguments in the database statement (104). For example,* ***the argument type*** *for each argument in the database statement can be passed to the optimizer. The following are examples of such argument types: literals,* ***bind variables****, columns,* ***type attributes****, NULL , none of the above. This is not intended to be an exhaustive list of possible descriptions that can be passed to the optimizer. Other descriptions regarding the characteristics, structure, or type of the argument can be used within the scope of the invention. The description of the arguments can be used in the present invention to more effectively estimate selectivity and cost, particularly for database statements containing variables whose values are unknown at compile-time;* Agarwal Column 5 Lines 50-61, i.e., *This database statement queries for all entries from table emp_table where the value of column "Table2.col" equals the bind variable :x.* ***The arguments to the "equal()" function are passed to the optimizer*** *(i.e., the "Table2.col" and ":x" arguments).* ***Since :x is a bind variable whose value is unknown at compile-time****, the present invention further passes a description of the arguments to the optimizer. For example,* ***the argument "type" of the arguments can be passed to the optimizer****. The argument type of the Table2.col argument is "column" and the argument type of the :x argument is "bind variable")*

"when there is a match between the data in the application structure and data in the optimization information in the bind-in structure, executing the statement with the optimization information to perform one of fetching data from the data store (and inserting data into the data store)" (Agarwal, Column 4 Lines 41-44, i.e., *The optimizer then selects for execution plan having the lowest relative cost*) "to perform one of fetching data from the data store (and inserting data into the data store) (Agarwal, Column 4 Line 41 through Column 5 Line 45, i.e., *As an example, consider the following database statement:*

***SELECT ****

*FROM Table1 , table"* (column 4 lines 45-47) )

"wherein the bind-out structure and the statement have a same section number" (Agarwal in view Kaluskar and further in view of Hayes as applied to claim 1 above).

"when there is not a match between the data in the application structure and the optimization information" (Agarwal (Column 4 Lines 41-44) in view of Kaluskar, Column 3 Line 57-64, i.e., ***If a match is not found***, *then compilation proceeds as in Fig. 1* ; Also note Kaluskar Column 3 Lines 15-25, i.e., *The result of a hard parse is* ***a memory-resident data structure***, *which dictates to the server, by way of the execution plan,* ***the best method for carrying out the database statement request***. *A cursor is one example of such a data structure that is essentially a handle to a memory location where the details and results of a parsed and optimized database statement reside. The cursor comprises, among other things, a compiled SQL statement with its associated execution plan;* Kaluskar, Column 3 Line 39-45, i.e., *The systems and methods for*

*sharing of execution plans for similar SQL statements overcome the disadvantages discussed above* ***by reusing the execution plan*** *of an existing cursor in those situations* ***where a client issues a SQL statement similar to another SQL statement previously compiled*** *;* Kaluskar, column 3 Line 51-55, i.e., *Step 205 computes* ***a hash value*** *to determine* ***if a matching SQL statement exists in the shared memory pool****. If a match is found, step 210, then the performance penalty of initiating a hard parse can be avoided altogether--the existing plan compiled for the matching SQL statement will be reused for the instant SQL statement (step 220). If a match is not found, then compilation proceeds as in FIG. 1 (step 215), beginning with parse phase 150 and ending with execution plan creation 130.* Note that Kaluskar's method is employs "a hash" value to match the currently issued SQL statement with the SQL statement that have already been compiled, and has been associated with a cursor which points to the best execution plan for that existing (i.e., already compiled) SQL optimization information )

"regenerating optimization information"" (Kaluskar, Column 3 Line 57-64, i.e., *If a match is not found, then compilation proceeds as in FIG. 1 (step 215), beginning with parse phase 150 and* ***ending with execution plan creation 130***);

"executing the statement with the regenerated optimization information to perform one of fetching data from the data store and inserting data into the data store" (Kaluskar, Figure 2, i.e., 210 –N→ COMPILE CURSOR AND CREATE EXECUTION PLAN (Figure 1) ---→ the point after REUSE CURSOR 220; Kaluskar, Column 3 Line 39-45, i.e., *The systems and methods for sharing of execution plans for similar SQL*

*statements overcome the disadvantages discussed above* ***by reusing the execution plan*** *of an existing cursor in those situations where a client issues a SQL statement similar to another SQL statement previously compiled.* Particularly note the recursive loop of execution in the method of Kaluskar, that is, if there is not optimization information (i.e., execution plan) for a particular SQL and related SQL data, a new execution plan with optimization is regenerated for use later for executing the statement). Note that SQL statements are for fetching (SQL command: SELECT), updating (SQL command: UPDATE), and inserting (SQL command: INSERT) to fetch data from or store data into data stores or update or change data in data stores, which is notoriously well known to everyone in the art.

Claim 12 is rejected on the same basis claim 3.

Claim 18 is rejected on the same basis as claim 9.

Claim 19 is essentially the same as claim 1 except that it set forth the claimed invention as a article of manufacture rather than a method and rejected for the same reasons as applied hereinabove.

Claim 20 is essentially the same as claim 2 except that it set forth the claimed invention as a article of manufacture rather than a method and rejected for the same reasons as applied hereinabove.

Claim 21 is essentially the same as claim 3 except that it set forth the claimed invention as a article of manufacture rather than a method and rejected for the same reasons as applied hereinabove.

Claim 27 is essentially the same as claim 9 except that it set forth the claimed invention as a article of manufacture rather than a method and rejected for the same reasons as applied hereinabove.

Claim 28 is essentially the same as claim 10 except that it set forth the claimed invention as a article of manufacture rather than a method and rejected for the same reasons as applied hereinabove.

Claim 30 is essentially the same as claim 12 except that it set forth the claimed invention as a article of manufacture rather than a method and rejected for the same reasons as applied hereinabove.

Claim 36 is essentially the same as claim 18 except that it set forth the claimed invention as a article of manufacture rather than a method and rejected for the same reasons as applied hereinabove.

Claim 37 is essentially the same as claim 1 except that it set forth the claimed invention as a system rather than a method and rejected for the same reasons as applied hereinabove.

Claim 38 is essentially the same as claim 10 except that it set forth the claimed invention as a system rather than a method and rejected for the same reasons as applied hereinabove.

10. Claim 5, 14, 23, and 32 are rejected under 35 U.S.C. 103(a) as being unpatentable over Agarwal in view of Kaluskar and further in view of Hayes and further in view of Desai et al., (hereinafter "Desai", U.S. Patent Number 6567816).

Referring to claim 5, Agarwal in view of Kaluskar and further in view of Hayes does not explicitly disclose the limitations: "for fixed length data, storing an increment length by which a data pointer that is pointing to data in an application program area is to be incremented to find a location of an next data value and calculating the location of the next data value by adding the increment length to the data pointer."

Desai teaches the limitations:

"for fixed length data, storing an increment length by which a data pointer that is pointing to data in an application program area is to be incremented to find a location of an next data value and calculating the location of the next data value by adding the increment length to the data pointer" (Column 5 Line 34-49). Desai teaches a method and system for extracting data from database records, wherein offsets from the starting of the row in memory are used to determine corresponding column name by adding to the said column offset the length of fixed columns (Column 5 Line 34-49).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to modify the method of Agarwal in view of Kaluskar and further in view of Hayes to add the feature of employing offsets (increments) to locate (point to) next column (item/record) as taught by Desai to the method and system taught by Agarwal in view of Kaluskar and further in view of Hayes so that the resultant method would constitute the method of claim 1, which further comprises, for fixed length data, storing an increment length by which a data pointer that is pointing to data in an application program area is to be incremented to find a location of a next data value and calculating the location of the next data value by adding the increment length to the data pointer. One would have been motivated to do so simply to locate a memory location, which is well known in the art.

Claim 14 is rejected on the same basis as claim 5.

Claim 23 is essentially the same as claim 5 except that it set forth the claimed invention as a article of manufacture rather than a method and rejected for the same reasons as applied hereinabove.

Claim 32 is essentially the same as claim 14 except that it set forth the claimed invention as a article of manufacture rather than a method and rejected for the same reasons as applied hereinabove.

11. Claim 6-8, 15-17, and 24-26, and 33-35 are rejected under 35 U.S.C. 103(a) as being unpatentable over Agarwal in view of Kaluskar and further in view of Hayes and further in view of Jordan II et al. (hereinafter "Jordan", U.S. Patent Number 5875442).

Referring to claim 6, Agarwal in view of Kaluskar and further in view of Hayes does not explicitly disclose the limitation: "for distributed computing, at a client computer, calculating a location of data in a client communications buffer".

Jordan teaches the limitation:

"for distributed computing, at a client computer, calculating a location of data in a client communications buffer" (Jordan II, Figure 3 and Column 4 Line 12-29). Jordan teaches a method and system for accessing a remote database, wherein location of data in communication buffer are calculated (Jordan II, Figure 3 and Column 4 Line 12-29).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to modify the method of Agarwal in view of Kaluskar and further in view of Hayes add the feature of calculating memory location in a communication buffer as taught by Jordan II. et al. to the method of Agarwal in view of Kaluskar and further in view of Hayes as applied to claim 1 above so that the resultant method would further comprise, for distributed processing, at a client computer, calculating a location of data in a client communications buffer. One would have been motivated to do so in order to *provide dynamic buffering to enhance a database server*" (Jordan, Column 1 Line 26-29).

As per claim 7, Agarwal in view of Kaluskar and further in view of Hayes and further in view of Jordan II teaches the limitation:

"for distributed processing, at a server computer, calculating a location of data in a server communication buffer" (Jordan II, Figure 3 and Column 4 Line 12-29).

As per claim 8, Agarwal in view of Kaluskar and further in view of Hayes and further in view of Jordan II teaches the limitation:

"for distributed processing, at a client computer, calculating a location of data in a an application address space" (Jordan II, Figure 3 and Column 4 Line 12-29).

Claims 15-17 are rejected on the same basis claim 6-8 respectively.

Claims 24-26 are rejected on the same basis claim 6-8 respectively.

Claims 33-35 are rejected on the same basis claim 6-8 respectively.

## *Response to Arguments*

12. The applicant's arguments filed on September 28, 2007, have been fully considered but are moot in view of the new ground(s) of rejection.

## *Conclusion*

13. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure are as follows.

(1) Srinivasan et al., (U.S. Patent Number 5893104)

(2) Agarwal et al., (U.S. Patent Number 6401083)

(3) Agarwal et al., (U.S. Patent Number 6370522).

14. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

## *Contact Information*

15. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Dennis Myint whose telephone number is (571) 272-5629. The examiner can normally be reached on 8:30AM-5:30PM Monday-Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Breene can be reached on (571) 272-4107. The fax phone number for the organization where this application or proceeding is assigned is 571-273-5629.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Cam Y Truong
primary Examiner

Dennis Myint
Examiner
AU-2162